CDF
Data Handling

J. Tseng
MIT/FNAL

4 March 2003

- ▷ Introduction
- ▷ dCache
- ▷ SAM
- ▷ Conclusion

Introduction

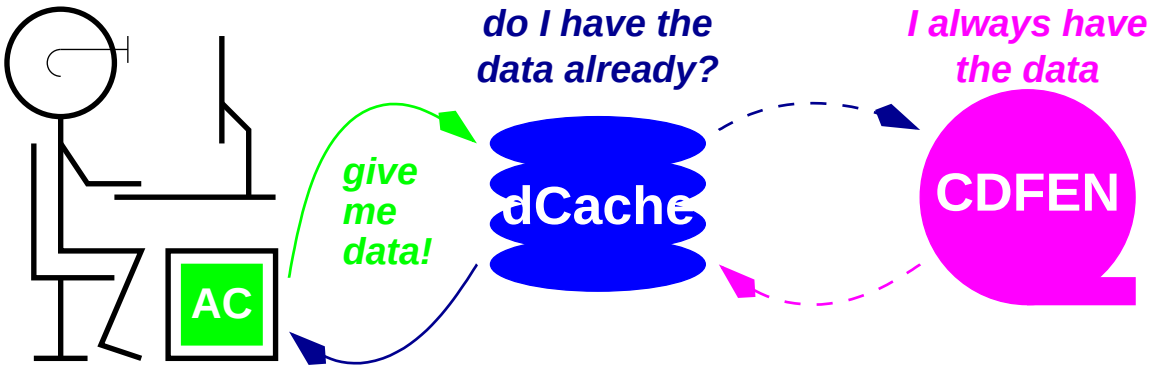
1. Browse datasets with DB Browser
   Data Catalog: Datasets' Registry Entries
2. Talk to `DHInput`:

```
talk DHInput
  include dataset hbhd08
exit
begin
```

▷ We try to keep modifications to the above to a minimum

dCache

What dCache Does



Benefits of dCache

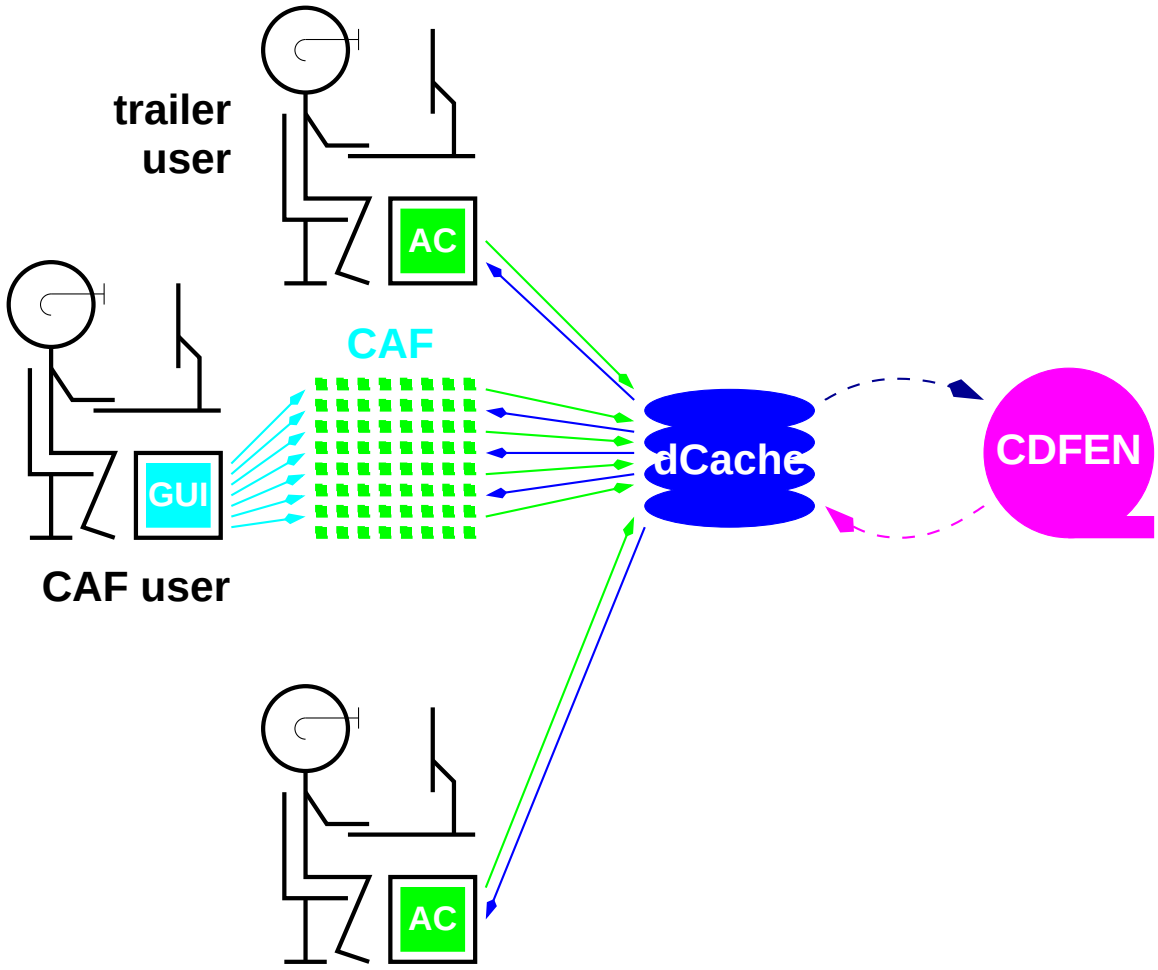
- ▷ automatic loading of requested files from tape (robust against single fileserver failures)
- ▷ optimize tape mounts (keep data on disk)
- ▷ spread files across fileservers
- ▷ ROOT I/O: multi-branch Edm, ntuples

dCache (II)

Benefits of dCache (continued)

▷ network-attached functionality

⇒ CAF, trailers, fcdfsgi2, remote



dCache Status

▷ 60TB in user system
(22TB currently in separate test stand
for disk testing)

▷ Improvements since January:

√ client software: reduce load 10 to 30×
(collaborators very helpful in using hpt2)

√ server configuration and debugging

√ server retries in case all else fails

• Many common failure modes resolved

◦ Still some left to fix
some peculiar to very heavy load
server retries successful

▷ Used dCache to load static fileservers

▷ Used by consumers of non-static datasets

♣ Caveat: light load

Lessons

- ▷ Single points of failure will be exploited
- ▷ Failures are superproportional to system scale
- ▷ "Offline" infrastructure systems are really real-time (*i.e.*, "online") systems
- ▷ Always have a backup system
- ▷ Conditions for "in-production" status:
  1. production quality disk systems
  2. scale up slowly: try to hit one scalability problem at a time
     (currently 150 CAF processes allowed)

SAM

"Sequential Access via Metadata"

DH with advanced functionality

▷ Flexible, journaled dataset management

- ◦ Move beyond 6-character mnemonics

→ more amenable to MC cornucopia

- ◦ Tracks history, parentage of data files, datasets
- ◦ Enhanced user ability to create/validate new, shareable datasets
  (MC, strips, ntuples)
- ◦ Archived user snapshots of growing datasets

→ users can reproduce their analyses' inputs

- ◦ Enables integration of partial reprocessing

→ fewer incantations for users to remember

SAM (II)

▷ Tracks successfully processed files
(also tracks crashes)

▷ Traffic shaping: less idling on CAF waiting for files to be staged to disk

▷ Integrate remote institutions' computing/storage

- ◦ MC generation
- ◦ analysis
- ◦ incorporate LHC clusters...

▷ Improved management of static fileservers
(clients not required to use SAM to read)

• SAM provides functionality useful for large datasets on large clusters

⇒ useful (later necessary) to CDF users

SAM for Summer

- Most functionality exists now "in test"
- Advanced functionality will become more useful as data grows
- DØ stresses SAM heavily, but probably not as much as CDF can

⇒ Introduce SAM as additional functionality, as needed by users, alongside dCache and other methods

For summer conferences:
SAM for MC generation, import from off-site

▷ large MC needs: more species, more of each

▷ mobilize large off-site clusters

▷ large operational load to generate, import

→ with SAM, could be operated by one person

Conclusion

▷ Many lessons learned about dCache

→ improved recent operation, albeit light load

▷ Many lessons learned from dCache

→ keep backup methods ready

▷ SAM has many benefits for CDF

- ◦ flexible, journaled dataset management
- ◦ traffic shaping for better CAF operation
- ◦ integration of remote computing/storage resources

▷ introduce SAM as additional functionality

▷ SAM for summer:
more MC of more kinds, from more sites